


```
CCCCCCCCC HH HH MM MM HH HH LL PPPPPPP KK KK PPPPPPP DDDDDDD DD
CCCCCCCCC HH HH MM MM HH HH LL PPPPPPP KK KK PPPPPPP DDDDDDD DD
CC CC HH HH MMMM MMMM HH HH LL PP PP KK KK PP PP DD DD
CC CC HH HH MMMM MMMM HH HH LL PP PP KK KK PP PP DD DD
CC CC HH HH MM MM MM HH HH LL PP PP KK KK PP PP DD DD
CC CC HHHHHHHHHH MM MM HHHHHHHHHH LL PP PP KKKKKK PPPPPPP DD DD
CC CC HHHHHHHHHH MM MM HHHHHHHHHH LL PP PP KKKKKK PPPPPPP DD DD
CC CC HH HH MM MM HH HH LL PP PP KK KK PP PP DD DD
CC CC HH HH MM MM HH HH LL PP PP KK KK PP PP DD DD
CC CC HH HH MM MM HH HH LL PP PP KK KK PP PP DD DD
CC CC HH HH MM MM HH HH LL PP PP KK KK PP PP DD DD
CCCCCCCCC HH HH MM MM HH HH LLLLLLLLLL PP PP KK KK PP PP DD DD
CCCCCCCCC HH HH MM MM HH HH LLLLLLLLLL PP PP KK KK PP PP DD DD
.....

LL IIIIII SSSSSSSS
LL IIIIII SSSSSSSS
LL II SS
LL II SS
LL II SS
LL II SSSSSS
LL II SSSSSS
LL II SS
LL II SS
LL II SS
LL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS
```



```
0001 0 %TITLE 'EDT$CHMHLPKPD - keypad help'
0002 0 MODULE EDT$CHMHLPKPD (
0003 0 IDENT = 'V04-000'
0004 0 ) =
0005 1 BEGIN
0006 1
0007 1 *****
0008 1 *
0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0011 1 * ALL RIGHTS RESERVED.
0012 1 *
0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0018 1 * TRANSFERRED.
0019 1 *
0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0022 1 * CORPORATION.
0023 1 *
0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0026 1 *
0027 1 *
0028 1 *****
0029 1
0030 1
0031 1 ++
0032 1 FACILITY: EDT -- The DEC Standard Editor
0033 1
0034 1 ABSTRACT:
0035 1
0036 1 This module handles keypad mode help.
0037 1
0038 1 ENVIRONMENT: Runs at any access mode - AST reentrant
0039 1
0040 1 AUTHOR: Bob Kushlis, CREATION DATE: Unknown
0041 1
0042 1 MODIFIED BY:
0043 1
0044 1 1-001 - Original. DJS 04-Feb-1981. This module was created by
0045 1 extracting the routine EDT$KPAD_HLP from the module CHANGE.BLI.
0046 1 1-002 - Regularize headers. JBS 02-Mar-1981
0047 1 1-003 - Use new message codes. JBS 04-Aug-1981
0048 1 1-004 - Add return values. JBS 02-Oct-1981
0049 1 1-005 - Revise the mapping of keys into help text. SMB 19-Nov-1981
0050 1 1-006 - Add a message for no help on nondefinable key. SMB 20-Nov-1981
0051 1 1-007 - Bypass EDT$CHM_HLP by calling EDT$SOUT_HLP directly. JBS 03-Jun-1982
0052 1 1-008 - Send a shift in on exit from this module. SMB 29-Jul-1982
0053 1 1-009 - New implementation of defined keys. JBS 13-Aug-1982
0054 1 1-010 - Don't add K_GOLD_BASE to the key, just look it up. JBS 16-Aug-1982
0055 1 1-011 - Allow for 8-bit keyboards, and fix a bug in lower case conversion. JBS 17-Aug-1982
0056 1 1-012 - Add SS3, for 8-bit keyboards. JBS 20-Aug-1982
0057 1 1-013 - Send out the shift-in sequence dependant on terminal. STS 27-Aug-1982
```

EDT\$CHMHLPKPD
V04-000

EDT\$CHMHLPKPD - keypad help

L 12

15-Sep-1984 23:59:50
14-Sep-1984 12:22:33

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]CHMHLPKPD.BLI;1

Page 2
(1)

```
: 58      0058 1 | 1-014 - Change the call to erase the screen. JBS 23-Oct-1982
: 59      0059 1 | 1-015 - Make sure the cursor is positioned properly before erasing the screen. JBS 01-Nov-1982
: 60      0060 1 | 1-016 - On return, rebuild the screen data base. JBS 01-Nov-1982
: 61      0061 1 | 1-017 - Add conditional for VT220 support. JBS 11-Feb-1983
: 62      0062 1 | 1-018 - Take out unnecessary calls to EDT$$OUT_FMTBUF. SMB 23-Feb-1983
: 63      0063 1 | 1-019 - Do the setting of the scrolling region for the keypad picture in code instead of in the help text.
: 64      0064 1 |      This way EDT does not lose track of the scrolling region. JBS 22-Mar-1983
: 65      0065 1 | 1-020 - Make sure we are in replace mode, so the help screen does not paint slowly. JBS 21-Apr-1983
: 66      0066 1 | --
: 67      0067 1 |
```

```

: 69      0068 1 %SBTTL 'Declarations'
: 70      0069 1
: 71      0070 1 : TABLE OF CONTENTS:
: 72      0071 1 :
: 73      0072 1
: 74      0073 1 REQUIRE 'EDT$SRC:TRAROUNAM';
: 75      0512 1
: 76      0513 1 FORWARD ROUTINE
: 77      0514 1     EDT$KPAD_HLP;
: 78      0515 1
: 79      0516 1
: 80      0517 1 : INCLUDE FILES:
: 81      0518 1 :
: 82      0519 1
: 83      0520 1 REQUIRE 'EDT$SRC:EDTREQ';
: 84      0655 1
: 85      0656 1 LIBRARY 'EDT$SRC:KEYPADDEF';
: 86      0657 1
: 87      0658 1 LIBRARY 'EDT$SRC:SUPPORTS';
: 88      0659 1
: 89      0660 1 :
: 90      0661 1 : MACROS:
: 91      0662 1 :
: 92      0663 1 :     NONE
: 93      0664 1 :
: 94      0665 1 : EQUATED SYMBOLS:
: 95      0666 1 :
: 96      0667 1 :
: 97      0668 1 BIND
: 98      L 0669 1     SHIFT_IN_100 = UPLIT (%STRING (
: 99      L 0670 1         %CHAR (ASC_K_SI),
: 100     L 0671 1         %CHAR (ASC_K_ESC), '[m')),
: 101     L 0672 1     SHIFT_IN_52 = UPLIT (%STRING (
: 102     L 0673 1         %CHAR (ASC_K_SI),
: 103     L 0674 1         %CHAR (ASC_K_ESC), 'G'));
: 104     0675 1
: 105     0676 1 LITERAL
: 106     0677 1     SI_LEN_52 = 3,
: 107     0678 1     SI_LEN_100 = 4;
: 108     0679 1
: 109     0680 1 :
: 110     0681 1 : OWN STORAGE:
: 111     0682 1 :
: 112     0683 1 :     NONE
: 113     0684 1 :
: 114     0685 1 : EXTERNAL REFERENCES:
: 115     0686 1 :
: 116     0687 1 :     In the routine

! Keypad mode help processor

! Shift in
! Video attributes off
! shift in
```



```
118 0688 1 %SBTTL 'EDT$$KPAD_HLP - keypad help'
119 0689 1
120 0690 1 GLOBAL ROUTINE EDT$$KPAD_HLP ! Keypad help
121 0691 1 =
122 0692 1
123 0693 1 ++
124 0694 1 FUNCTIONAL DESCRIPTION:
125 0695 1
126 0696 1 This routine handles help while in keypad mode. The help processor
127 0697 1 is first called to put up the keypad diagram. Then we go into a loop
128 0698 1 accepting characters and, if they are definable keys, we give the help
129 0699 1 message for the key.
130 0700 1
131 0701 1 FORMAL PARAMETERS:
132 0702 1
133 0703 1 NONE
134 0704 1
135 0705 1 IMPLICIT INPUTS:
136 0706 1
137 0707 1 EDT$$G_TI_TYP
138 0708 1
139 0709 1 IMPLICIT OUTPUTS:
140 0710 1
141 0711 1 EDT$$G_SCR_CHGD
142 0712 1 EDT$$G_CS_CNO
143 0713 1 EDT$$G_SCR_REBUILD
144 0714 1
145 0715 1 ROUTINE VALUE:
146 0716 1
147 0717 1 1 = ok, 0 = end of journal file
148 0718 1
149 0719 1 SIDE EFFECTS:
150 0720 1
151 0721 1 NONE
152 0722 1
153 0723 1 --
154 0724 1
155 0725 2 BEGIN
156 0726 2
157 0727 2 EXTERNAL ROUTINE
158 0728 2 EDT$$FMT_LIT, ! Format a literal
159 0729 2 EDT$$FMT_CRLF, ! Format a CRLF
160 0730 2 EDT$$OUT_FMTBUF, ! Dump the format buffer
161 0731 2 EDT$$OUT_HLP, ! Get help
162 0732 2 EDT$$TRN_KPADK, ! Translate an escape sequence
163 0733 2 EDT$$SC_POSABS : NOVALUE, ! Position the cursor
164 0734 2 EDT$$SC_ERAALL, ! Erase to end of screen
165 0735 2 EDT$$MSG_TOSTR, ! Get message text
166 0736 2 EDT$$TI_INPCH, ! Get a character of input
167 0737 2 EDT$$TST_BADK, ! Test for an undefinable key
168 0738 2 EDT$$SC_SETSCLLREG : NOVALUE, ! Set the scrolling region
169 0739 2 EDT$$SC_REP_MODE : NOVALUE; ! Make sure we are in replace (not insert) mode.
170 0740 2
171 0741 2 EXTERNAL
172 0742 2 EDT$$G_TI_TYP, ! Terminal type.
173 0743 2 EDT$$G_SCR_CHGD, ! 1 = screen update must rebuild the screen
174 0744 2
```

```
: 175      L 0745 2 %IF SUPPORT_VT220
: 176      0746 2 %THEN
: 177      0747 2     EDT$$B_CHAR_INFO : BLOCKVECTOR [256, 1, BYTE], ! Information about characters
: 178      0748 2 %FI
: 179      0749 2
: 180      0750 2     EDT$$G_CS_LNO, ! Line number to erase from
: 181      0751 2     EDT$$G_SCR_REBUILD, ! 1 = rebuild the screen data base
: 182      0752 2     EDT$$G_INSERT_MODE; ! 1 = screen is in insert mode
: 183      0753 2
: 184      0754 2     MESSAGES ((TORETKEY, FORHLPANO, TOEXITHLP, KEYNOTDEF));
: 185      0755 2
: 186      0756 2     LOCAL
: 187      0757 2     CURSOR_SAVED,
: 188      0758 2     HELP_STRING : BLOCK [CH$ALLOCATION (16)],
: 189      0759 2     CH,
: 190      0760 2     C;
: 191      0761 2
: 192      0762 2 !+
: 193      0763 2 !- Get the right terminal type.
: 194      0764 2 !-
: 195      0765 2
: 196      0766 2     IF (.EDT$$G_TI_TYP EQL TERM_VT100) !
: 197      0767 2     THEN
: 198      0768 2     EDT$$CPY_MEM (15, UPLIT (%STRING ('KEYPAD VT100 ')), HELP_STRING)
: 199      0769 2     ELSE
: 200      0770 2     EDT$$CPY_MEM (15, UPLIT (%STRING ('KEYPAD VT52 ')), HELP_STRING);
: 201      0771 2
: 202      0772 2 !+
: 203      0773 2 !- Because EDT$$OUT_HLP prints a blank line before the text it finds, for the keypad
: 204      0774 2 !- diagram that text starts with a sequence to erase the whole screen and position the
: 205      0775 2 !- cursor to the upper left corner. Therefore we need not do cursor positioning here.
: 206      0776 2 !- However, we do set the scrolling region to the full screen.
: 207      0777 2 !-
: 208      0778 2     CURSOR_SAVED = .EDT$$G_CS_LNO;
: 209      0779 2     EDT$$G_CS_LNO = 0;
: 210      0780 2     EDT$$G_SCR_CHGD = 1;
: 211      0781 2     EDT$$SC_SETSCLLREG (0, 24); ! Full-screen scrolling region
: 212      0782 2
: 213      0783 2     IF (.EDT$$G_INSERT_MODE NEQ 0) THEN EDT$$SC_REP_MODE (); ! Make sure we are in replace mode
: 214      0784 2
: 215      0785 2 !+
: 216      0786 2 !- Call help to get the keypad diagram.
: 217      0787 2 !-
: 218      0788 2     EDT$$OUT_HLP (HELP_STRING, 12, 0);
: 219      0789 2 !+
: 220      0790 2 !- Loop, displaying help on keypad keys.
: 221      0791 2 !-
: 222      0792 2
: 223      0793 2     WHILE 1 DO
: 224      0794 2     BEGIN
: 225      0795 2 !+
: 226      0796 2 !- Send a precautionary shift-in plus turn video attributes off
: 227      0797 2 !- (mostly in case the person issues a CTRL-O).
: 228      0798 2 !-
: 229      0799 2
: 230      0800 4     IF (.EDT$$G_TI_TYP EQL TERM_VT100) !
: 231      0801 3     THEN
```



```
232      0802      3      EDT$FMT_LIT (SHIFT_IN_100, SI_LEN_100)
233      0803      ELSE
234      0804      EDT$FMT_LIT (SHIFT_IN_52, SI_LEN_52);
235      0805
236      0806      !+
237      0807      !- Get next character.
238      0808
239      0809
240      0810      IF (EDT$TI_INPCH (C) EQL 0) THEN RETURN (0);
241      0811
242      0812      !+
243      0813      !- Convert lower case to upper.
244      0814
245      0815
246      L 0816      %IF SUPPORT_VT220
247      0817      %THEN
248      0818
249      0819      IF .EDT$B_CHAR_INFO [.C, 0, 0, 1, 0] THEN C = .C - 32;
250      0820
251      U 0821      %ELSE
252      0822
253      U 0823      IF ((.C GEQ %C'a') AND (.C LEQ %C'z')) THEN C = .C - %C'a' + %C'A';
254      U 0824
255      0825      %FI
256      0826
257      0827      !+
258      0828      !- Erase the screen.
259      0829
260      0830      EDT$G_CS_LNO = 0;
261      0831      EDT$G_SCR_CHGD = 1;
262      0832      EDT$SC_POSABS (0, 0);
263      0833      EDT$SC_ERAALL ();
264      0834      !+
265      0835      !- When we return to change mode we must rebuild the screen and the screen data base.
266      0836
267      0837      EDT$G_SCR_CHGD = 1;
268      0838      EDT$G_SCR_REBUILD = 1;
269      0839      EDT$G_CS_LNO = .CURSOR_SAVED;
270      0840      !+
271      0841      !- If the character is a space, exit the loop
272      0842      !- and return to the original editing position.
273      0843
274      0844
275      0845      IF (.C EQL ASC_K_SP) THEN EXITLOOP;
276      0846
277      0847      !+
278      0848      !- If the character is an escape, CSI or SS3, translate the escape or control sequence.
279      0849
280      0850
281      L 0851      %IF SUPPORT_VT220
282      0852      %THEN
283      0853
284      0854      IF ((.C EQL ASC_K_ESC) OR (.C EQL ASC_K_CSI) OR (.C EQL ASC_K_SS3)) THEN EDT$TRN_KPADK (C);
285      0855
286      U 0856      %ELSE
287      U 0857
288      U 0858      IF (.C EQL ASC_K_ESC) THEN EDT$TRN_KPADK (C);
```



```
289      U 0859 3
290      0860 3 %FI
291      0861 3
292      0862 3
293      0863 3 !+ If the character is a carriage return, re-display the keypad diagram.
294      0864 3 !-
295      0865 3
296      0866 4 IF (.C EQL ASC_K_CR)
297      0867 3 THEN
298      0868 3 EDT$$OUT_HLP (HELP_STRING, 12, 0)
299      0869 3 ELSE
300      0870 4 BEGIN
301      0871 4 !+
302      0872 4 !- Give a different message if the key is not definable.
303      0873 4
304      0874 4
305      0875 5 IF (EDT$$TST_BADK (.C) AND EDT$$TST_BADK (.C + K_GOLD_BASE))
306      0876 5 THEN
307      0877 5 BEGIN
308      0878 5 !+
309      0879 5 !- The key is not defineable; display a message but continue looping.
310      0880 5
311      0881 5 EDT$$MSG_TOSTR (EDT$_KEYNOTDEF);
312      0882 5 EDT$$FMT_CRLF ();
313      0883 5 END
314      0884 4 ELSE
315      0885 5 BEGIN
316      0886 5 !+
317      0887 5 !- Get the key number so we can create the topic (key number is 3 digits: 000-499)
318      0888 5
319      0889 5 CH$WCHAR (.C/100 + %C'0', CH$PTR (HELP_STRING, 13));
320      0890 5 CH = (.C MOD 100);
321      0891 5 CH$WCHAR (.CH/10 + %C'0', CH$PTR (HELP_STRING, 14));
322      0892 5 CH$WCHAR (.CH MOD 10 + %C'0', CH$PTR (HELP_STRING, 15));
323      0893 5 !+
324      0894 5 !- Get the help text for the key. The GOLD meaning of the key is included.
325      0895 5
326      0896 5 EDT$$OUT_HLP (HELP_STRING, 16, 0);
327      0897 4 END;
328      0898 4
329      0899 4 !+
330      0900 4 !- Output the help instructions.
331      0901 4
332      0902 4 EDT$$FMT_CRLF ();
333      0903 4 EDT$$MSG_TOSTR (EDT$_TORETKEY);
334      0904 4 EDT$$FMT_CRLF ();
335      0905 4 EDT$$MSG_TOSTR (EDT$_TOEXITHLP);
336      0906 4 EDT$$FMT_CRLF ();
337      0907 4 EDT$$MSG_TOSTR (EDT$_FORHLPANO);
338      0908 4 EDT$$OUT_FMTBUF ();
339      0909 3 END;
340      0910 3
341      0911 2 END;
342      0912 2
343      0913 2 !+
344      0914 2 !- Again be sure we have the none video terminal attributes in case
345      0915 2 ! this module terminates abnormally.
```

EDT\$CHMHLPKPD
V04-000

EDT\$CHMHLPKPD - keypad help
EDT\$\$KPAD_HLP - keypad help

E 13
15-Sep-1984 23:59:50
14-Sep-1984 12:22:33

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]CHMHLPKPD.BLI;1

Page 8
(3)

```

: 346      0916  2  !-
: 347      0917  2
: 348      0918  3
: 349      0919  2
: 350      0920  2
: 351      0921  2
: 352      0922  2
: 353      0923  2
: 354      0924  2
: 355      0925  2
: 356      0926  1

      IF (.EDT$$G_TI_TYP EQL TERM_VT100)
      THEN
        EDT$$FMT_LIT (SHIFT_IN_100, SI_LEN_100)
      ELSE
        EDT$$FMT_LIT (SHIFT_IN_52, SI_LEN_52);

      EDT$$OUT_FMTBUF ();
      RETURN (T);
      END;
```

! of routine EDT\$\$KPAD_HLP

```

      .TITLE EDT$CHMHLPKPD EDT$CHMHLPKPD - keypad help
      .IDENT \V04-000\

      .PSECT _EDT$CODE,NOWRT, SHR, PIC,2

      6D 5B 1B 0F 00000 P.AAA: .ASCII <15><27>\[m\
      00 47 1B 0F 00004 P.AAB: .ASCII <15><27>\G\<0>
20 20 20 30 30 31 54 56 20 44 41 50 59 45 4B 00008 P.AAC: .ASCII \KEYPAD VT100 \<0>
      00 00017
20 20 20 20 32 35 54 56 20 44 41 50 59 45 4B 00018 P.AAD: .ASCII \KEYPAD VT52 \<0>
      00 00027
```

```

      SHIFT_IN_100= P.AAA
      SHIFT_IN_52= P.AAB

      .EXTRN EDT$$FMT_LIT, EDT$$FMT_CRLF
      .EXTRN EDT$$OUT_FMTBUF
      .EXTRN EDT$$OUT_HLP, EDT$$STRN_KPADK
      .EXTRN EDT$$SC_POSABS, EDT$$SC_ERAALL
      .EXTRN EDT$$MSG_TOSTR, EDT$$TI_INPCH
      .EXTRN EDT$$TST_BADK, EDT$$SC_SETSCLLREG
      .EXTRN EDT$$SC_REP_MODE
      .EXTRN EDT$$G_TI_TYP, EDT$$G_SCR_CHGD
      .EXTRN EDT$$B_CHAR_INFO
      .EXTRN EDT$$G_CS_LNO, EDT$$G_SCR_REBUILD
      .EXTRN EDT$$G_INSERT_MODE
      .EXTRN EDT$TORETKEY, EDT$FORHLPANO
      .EXTRN EDT$TOEXITHLP, EDT$KEYNOTDEF
```

```

      OFFC 00000

      5B 00000000G 00 9E 00002
      5A 00000000G 00 9E 00009
      59 C5 AF 9E 00010
      58 00000000G 00 9E 00014
      57 00000000G 00 9E 0001B
      56 00000000G 00 9E 00022
      5E 14 C2 00029
      02 6B D1 0002C
      08 12 0002F
      04 AE 08 A9 0F 28 00031
      04 AE 18 A9 06 11 00037
      54 66 D0 0003F 1$:
      66 D4 00042 2$:

      .ENTRY EDT$$KPAD_HLP, Save R2,R3,R4,R5,R6,R7,R8,- R9,R10,R11
      MOVAB EDT$$G_TI_TYP, R11
      MOVAB EDT$$G_SCR_CHGD, R10
      MOVAB SHIFT_IN_100, R9
      MOVAB EDT$$FMT_CRLF, R8
      MOVAB EDT$$MSG_TOSTR, R7
      MOVAB EDT$$G_CS_LNO, R6
      SUBL2 #20, SP
      CMPL EDT$$G_TI_TYP, #2
      BNEQ 1$
      MOVOC3 #15, P.AAC, HELP_STRING
      BRB 2$
      MOVOC3 #15, P.AAD, HELP_STRING
      MOVL EDT$$G_CS_LNO, CURSOR_SAVED
      CLRL EDT$$G_CS_LNO
```


EDT\$CHMHLPKPD
V04-000

EDT\$CHMHLPKPD - keypad help
EDT\$KPAD_HLP - keypad help

F 13
15-Sep-1984 23:59:50
14-Sep-1984 12:22:33

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]CHMHLPKPD.BLI;1

Page 9
(3)

6A	01	D0	00044	MOVL	#1, EDT\$G_SCR_CHGD	0780
	18	DD	00047	PUSHL	#24	0781
	7E	D4	00049	CLRL	-(SP)	
00000000G 00	02	FB	0004B	CALLS	#2, EDT\$SSC_SETSCLLREG	
00000000G 00	00	D5	00052	TSTL	EDT\$G_INSERT_MODE	0783
	07	13	00058	BEQL	3\$	
00000000G 00	00	FB	0005A	CALLS	#0, EDT\$SSC_REP_MODE	
7E	0C	7D	00061	MOVQ	#12, -(SP)	0788
	0C	9F	00064	PUSHAB	HELP STRING	
00000000G 00	03	FB	00067	CALLS	#3, EDT\$OUT_HLP	
02	6B	D1	0006E	CMPL	EDT\$G_TI_TYP, #2	0800
	06	12	00071	BNEQ	5\$	
	04	DD	00073	PUSHL	#4	0802
	59	DD	00075	PUSHL	R9	
	05	11	00077	BRB	6\$	
	03	DD	00079	PUSHL	#3	0804
00000000G 00	A9	9F	0007B	PUSHAB	SHIFT IN 52	
	02	FB	0007E	CALLS	#2, EDT\$FMT_LIT	
00000000G 00	5E	DD	00085	PUSHL	SP	0810
	01	FB	00087	CALLS	#1, EDT\$TI_INPCH	
	50	D5	0008E	TSTL	R0	
	03	12	00090	BNEQ	7\$	
	011D	31	00092	BRW	18\$	
03 00 BE40	00	9E	00095	MOVAB	EDT\$B_CHAR_INFO, R0	0819
6E	00	E1	0009C	BBC	#0, a[R0], 8\$	
	20	C2	000A2	SUBL2	#32, C	
6A	66	D4	000A5	CLRL	EDT\$G_CS_LNO	0830
	01	D0	000A7	MOVL	#1, EDT\$G_SCR_CHGD	0831
	7E	7C	000AA	CLRQ	-(SP)	0832
00000000G 00	02	FB	000AC	CALLS	#2, EDT\$SSC_POSABS	
00000000G 00	00	FB	000B3	CALLS	#0, EDT\$SSC_ERAALL	0833
6A	01	D0	000BA	MOVL	#1, EDT\$G_SCR_CHGD	0837
00000000G 00	01	D0	000BD	MOVL	#1, EDT\$G_SCR_REBUILD	0838
66	54	D0	000C4	MOVL	CURSOR_SAVED, EDT\$G_CS_LNO	0839
20	6E	D1	000C7	CMPL	C, #32	0845
	03	12	000CA	BNEQ	9\$	
	00C1	31	000CC	BRW	15\$	
1B	6E	D1	000CF	CMPL	C, #27	0854
	12	13	000D2	BEQL	10\$	
0000009B 8F	6E	D1	000D4	CMPL	C, #155	
	09	13	000DB	BEQL	10\$	
0000008F 8F	6E	D1	000DD	CMPL	C, #143	
	09	12	000E4	BNEQ	11\$	
	5E	DD	000E6	PUSHL	SP	
00000000G 00	01	FB	000E8	CALLS	#1, EDT\$STRN_KPADK	
53	6E	D0	000EF	MOVL	C, R3	0866
0D	53	D1	000F2	CMPL	R3, #13	
	03	12	000F5	BNEQ	12\$	
	FF67	31	000F7	BRW	3\$	
	53	DD	000FA	PUSHL	R3	0875
00000000G 00	01	FB	000FC	CALLS	#1, EDT\$TST_BADK	
1C	50	E9	00103	BLBC	R0, 13\$	
	C3	9F	00106	PUSHAB	500(R3)	
00000000G 00	01	FB	0010A	CALLS	#1, EDT\$TST_BADK	
0E	50	E9	00111	BLBC	R0, 13\$	
	8F	DD	00114	PUSHL	#EDT\$ KEYNOTDEF	0881
67	01	FB	0011A	CALLS	#1, EDT\$MSG_TOSTR	

EDT\$CHMHLPKPD
V04-000

EDT\$CHMHLPKPD - keypad help
EDT\$\$KPAD_HLP - keypad help

G 13
15-Sep-1984 23:59:50
14-Sep-1984 12:22:33

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]CHMHLPKPD.BLI;1

Page 10
(3)

		68	00	FB	0011D	CALLS	#0, EDT\$\$FMT_CRLF	0882
			40	11	00120	BRB	14\$	0875
		53	00000064	8F	C7 00122	DIVL3	#100, R3, R0	0889
7E	11	50		30	81 0012A	ADDB3	#48, R0, HELP_STRING+13	
52		53		01	7A 0012F	EMUL	#1, R3, #0, -7SP)	0890
		8E	00000064	8F	7B 00134	EDIV	#100, (SP)+, CH, CH	
		52		0A	C7 0013D	DIVL3	#10, CH, R0	0891
7E	12	50		30	81 00141	ADDB3	#48, R0, HELP_STRING+14	
50		52		01	7A 00146	EMUL	#1, CH, #0, -7SP)	0892
		8E		0A	7B 0014B	EDIV	#10, (SP)+, R0, R0	
	13	50		30	81 00150	ADDB3	#48, R0, HELP_STRING+15	
		7E		10	7D 00155	MOVQ	#16, -(SP)	0896
			0C	AE	9F 00158	PUSHAB	HELP_STRING	
		00000000G	00	03	FB 0015B	CALLS	#3, EDT\$\$OUT_HLP	
		68		00	FB 00162	CALLS	#0, EDT\$\$FMT_CRLF	0902
			00000000G	8F	DD 00165	PUSHL	#EDT\$ TORETKEY	0903
		67		01	FB 0016B	CALLS	#1, EDT\$\$MSG_TOSTR	
		68		00	FB 0016E	CALLS	#0, EDT\$\$FMT_CRLF	0904
			00000000G	8F	DD 00171	PUSHL	#EDT\$ TOEXITHELP	0905
		67		01	FB 00177	CALLS	#1, EDT\$\$MSG_TOSTR	
		68		00	FB 0017A	CALLS	#0, EDT\$\$FMT_CRLF	0906
			00000000G	8F	DD 0017D	PUSHL	#EDT\$ FORHLPANO	0907
		67		01	FB 00183	CALLS	#1, EDT\$\$MSG_TOSTR	
		00000000G	00	00	FB 00186	CALLS	#0, EDT\$\$OUT_FMTBUF	0908
				FEDE	31 0018D	BRW	4\$	0793
		02		6B	D1 00190	CMPL	EDT\$\$G_TI_TYP, #2	0918
				06	12 00193	BNEQ	16\$	
				04	DD 00195	PUSHL	#4	0920
				59	DD 00197	PUSHL	R9	
				05	11 00199	BRB	17\$	
				03	DD 0019B	PUSHL	#3	0922
			04	A9	9F 0019D	PUSHAB	SHIFT IN 52	
		00000000G	00	02	FB 001A0	CALLS	#2, EDT\$\$FMT_LIT	
		00000000G	00	00	FB 001A7	CALLS	#0, EDT\$\$OUT_FMTBUF	0924
			50	01	D0 001AE	MOVL	#1, R0	0925
				04	001B1	RET		
				50	D4 001B2	CLRL	R0	0926
				04	001B4	RET		

; Routine Size: 437 bytes, Routine Base: _EDT\$CODE + 0028

; 357 0927 1
; 358 0928 1 !<BLF/PAGE>

EDT\$CHMHLPKPD
V04-000

EDT\$CHMHLPKPD - keypad help
EDT\$KPAD_HLP - keypad help

H 13
15-Sep-1984 23:59:50
14-Sep-1984 12:22:33

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]CHMHLPKPD.BLI;1

Page 11
(4)

: 360 0929 1 END
: 361 0930 1
: 362 0931 0 ELUDOM

! of module EDT\$CHMHLPKPD

PSECT SUMMARY

:
: Name Bytes Attributes
: _EDT\$CODE 477 NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
-\$255\$DUA28:[EDT.SRC]EDT.L32;1	377	8	2	40	00:00.2
-\$255\$DUA28:[EDT.SRC]PSECTS.L32;1	2	1	50	7	00:00.1
-\$255\$DUA28:[EDT.SRC]KEYPADDEF.L32;1	34	1	2	7	00:00.1
-\$255\$DUA28:[EDT.SRC]SUPPORTS.L32;1	2	1	50	5	00:00.1

COMMAND QUALIFIERS

:
: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACEBACK/LIS=LIS\$:CHMHLPKPD/OBJ=OBJ\$:CHMHLPKPD MSRC\$:CHMHLPKPD.BLI/UPDATE=(ENH\$:C
: HMHLPKPD)

: Size: 437 code + 40 data bytes
: Run Time: 00:20.4
: Elapsed Time: 00:24.9
: Lines/CPU Min: 2735
: Lexemes/CPU-Min: 8650
: Memory Used: 142 pages
: Compilation Complete

0131 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY